Computer Science and Application 计算机科学与应用, 2017, 7(3), 248-254
Published Online March 2017 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2017.73031

Hans汉斯

The Implementation of Video Player with Image Algorithm

Pengfei Xue, Tian Xia, Zhengzhi Li

College of Computer Science, Sichuan University, Chengdu Sichuan
Email: f79534672@gmail.com

Received: Mar. 11th, 2017; accepted: Mar. 26th, 2017; published: Mar. 29th, 2017

Abstract

With the development of computer science and Internet, video player has become an indispensable application of each computer terminal. Most player usually play video with storage in the form of digital signal. With the large variety, powerful function, players can be able to support the mainstream multimedia formats and have many powerful ways of decoding. But, most of them are not open source, and don't support the deep user-defined operation, and even have defects in Audio & Video synchronization and skip on progress. This paper designed a Qt-based player, which not only meets the demand of normal play but also can load user-defined image algorithm. It has jumping single frame and loading image algorithm and other video playback functions, and it also has saving frame sequence, processing video and other video processing functions.

Keywords

Video Play, Loading Image Algorithm, Saving Frame Sequence, Video Algorithm

动态加载图像算法的视频播放器的实现

薛鹏飞，夏　添，李政志

四川大学计算机学院，四川　成都
Email: f79534672@gmail.com

收稿日期：2017年3月11日；录用日期：2017年3月26日；发布日期：2017年3月29日

摘　要

在计算机与互联网高速发展的今天，视频播放器已经成为每台计算机终端必不可少的应用。现在的视频播放器通常指能播放以数字信号形式存储的视频的软件，种类繁多，功能强大，能够支持主流的多媒体

文章引用: 薛鹏飞, 夏添, 李政志. 动态加载图像算法的视频播放器的实现[J]. 计算机科学与应用, 2017, 7(3): 248-254.
https://doi.org/10.12677/csa.2017.73031

格式，具有多种强大的解码方式，播放效果清晰流畅。但是大多数视频播放器并不开源，不支持深层的用户自定义操作，功能仅仅局限于播放，并且播放效果单一，对于视频处理并未有涉及，部分播放器在音视频同步问题和进度条跳转问题上还存在缺陷。本文基于Qt平台设计出一款既能满足普通播放需求，又可以加载用户自定义图像算法以实现不同的播放效果的播放器控件，具有单帧跳转、加载图像算法播放等视频播放功能，同时具有保存帧序、视频算法处理等视频处理功能。

关键词

视频播放，加载图像算法，保存帧序，视频算法

1. 研究背景

在计算机与互联网高速发展的今天，视频播放器已经成为每台计算机终端必不可少的应用。现在的视频播放器通常指能播放以数字信号形式存储的视频的软件。在不同的领域，视频播放器发挥的作用不尽相同，起到的作用也或大或小。现有的大多数视频播放器，都携带相应的第三方解码器来还原经过压缩后的媒体文件，并且还要内置一套转换频率以及缓冲的算法。当然，大多数的视频播放器都支持音频播放。这些播放器都将重点集中在提高播放效果和播放效率，而无法体现多媒体丰富的可扩展性。

机器视觉作为近几年新兴并迅速发展的测量方式，已经开始逐渐取代传统人眼测量方式。在很多行业中也已经有了比较成熟的机器视觉的应用，如基于仪表板总成智能集成检测系统、金属板表面自动控伤系统、汽车车身检测系统、纸币印刷质量检测系统、智能交通管理系统、金相分析、医疗图像分析、瓶装啤酒生产流水线检测系统、大型工件平行垂直度测量仪螺纹钢外形轮廓尺寸检测系统、轴承实时监控系统、金属表面的裂纹测量等。

在这些现有的机器视觉监测系统中，现有的系统实现方式大多是通过对静态的图像进行捕捉并进行处理，这样不仅效率不高，在某些需要大量的上下文相关的图像序列中，非常难以实现图像的检测。如：气象业中对云层的运动监测系统则是需要对动态的图像数据进行分析处理。因此，视频处理算法已经逐渐走上这些行业的舞台，视频播放器中增加视频处理功能也显得十分的重要。

2. 音视频同步

由于多媒体音视频文件中数据源是以包的形式进行组织。大部分文件分为三个轨道：视频轨道、音频轨道和字幕轨道，这里重点讨论前两种轨道。对于不同的文件格式，甚至对于同一种文件格式的不同编码标准，音视频包的排列方式不尽相同。用 v 代表视频包，a 代表音频包，部分媒体格式文件的音视频包排列方式如图 1 所示。

程序在对解码后的音视频数据进行播放的时候，若是采用根据时钟的方式来限制解码速度，并将解码后的音视频包立即播放的方式，视频是可以正常播放，而音频数据由于不连续，在播放音频的时候会出现卡顿的现象。因此，仅仅使用对音视频包进行编号并计算播放时间后进行同步播放的方式并不可行，这就需要使用线程间通信的方式或者时间戳的方式来进行音视频同步。常见的音视频同步技术有基于缓冲区数据控制的同步[1]，基于播放时间的同步，多路复用同步技术，时间戳同步技术，同步信道技术，

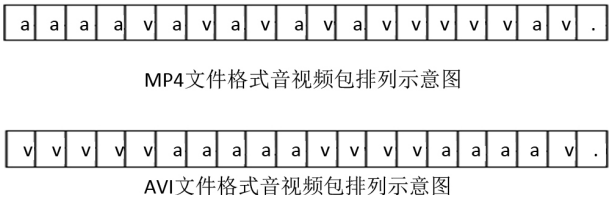

Figure 1. Arrangement of audio & video package
图 1. 音视频包排列方式

同步标记技术，基于反馈的同步技术，时间戳同步信道技术，基于 RTP 的同步技术，音频跟随视频的同步技术等等[2]。在此，这些技术的具体实现方式和优缺点不再熬述。

使用时间戳同步音视频的方式主要有三种：音频同步于视频，视频同步于音频，音频视频同步于系统时钟[3]。这三种方式各有优劣，下面分别介绍这三种方式的优缺点。

音频同步于视频同步方式不会出现由于视频同步问题而导致的丢帧或视频帧无限制播放的情况，但是由于这种方式需要视频给定标准始终，而视频的时钟是根据视频文件头的 fps (帧率)来确定的，对于某些不按照视频编码标准进行编码的视频流，会出现 fps 值不准确的情况，甚至出现 fps 值不存在的情况。并且，若音频不断的进行与视频进行同步操作，导致音频播放忽快忽慢，甚至出现卡顿现象，对播放效果造成很大的影响[4]。因此，这种方式并不能作为普遍的音视频同步方案。

音频视频同步于系统时钟的同步方式是音视频数据本身并不互相干涉，音频与视频的各自的时间戳只需要与系统地始终进行对比决定是否立即播放即可。这种方式对于大多数的音视频文件可行，但若出现极端情况，由于时间戳是线性增长的，一段数据流中，若大量的只出现音频流或大量的只出现视频流，会出现视频卡顿或者音频卡顿的情况。而且，假若音频时间戳与视频时间戳的时间基并不标准，会出现视频音频无法同步的现象。因此，大多数媒体播放器并不采取这种同步方式。

视频同步于音频方式是音频数据直接交给声卡播放，视频数据则根据其时间戳与正在播放的音频的时间戳进行对比，根据对比结果决定视频帧是否立即播放。由于这种方式不会出现音频卡顿的现象，而且无需调用系统时间而浪费资源，因此是比较好的音视频同步方案。

3. 非关键帧跳转

对于大多数的多媒体文件，为了减小视频文件占用空间的大小，在它们的视频流中，并不是每一个视频包中包含的帧数据都是一整幅的图像，而是只有一部分视频包中包含的是一整幅图像，另一部分视频包中只是相对于前一帧或者相对于附近几帧图像的变化信息，根据变化信息才计算出当前帧的完整图像。这些包含完整图像数据的帧称为关键帧，不包含图像完整信息而只包含图像变化信息的帧称为非关键帧。对于某些背景变化非常小的视频，整个视频文件中只包含一个关键帧，后面的帧都是根据前一帧计算得来的[5]。

在视频正常播放的时候，并不用关心这个问题。因为在解码的时候，只需向其提供一个全局变量，这个全局变量保存的就是当前帧的完整图像信息，而在解码下一帧视频的时候会自动根据上一次帧的信息来计算出当前帧的完整图像。

在具有视频进度跳转功能的视频播放器中，用户会随机的跳转到整个视频文件的某一帧图像，而当这一帧图像为非关键帧时，继续利用上一次的解码结果进行计算当前帧的完整图像时，便会出现图像“花屏”现象，甚至出现图像数据错误的情况。

因此，在设计视频播放器跳转功能的时候，需要考虑如何计算非关键帧完整图像的问题。

现在大多数的本地视频播放器和网络播放器基本上都拥有进度条转功能，经过使用后发现，它们在

处理非关键帧跳转的问题上解决方式不尽相同。主要分为以下三种：

(1) 无论跳转至关键帧还是跳转到非关键帧，都继续使用上一帧的图像信息作为背景信息继续进行解码。这种方式的缺点很明显，就是在跳转到非关键帧时，由于跳转之前的图像信息与跳转之后非关键帧所对应的关键帧的图像信息很有可能不相同，因此会出现短暂的“花屏”问题，直至播放到下一个关键帧为止视频播放才能恢复正常。

这种解决方案由于会出现“花屏”问题，在现有的视频播放器中比较少见。

(2) 跳转到分关键帧时，读取该视频包的信息，判断是否为关键帧。若为关键帧，则只需将进度时间与读取到的视频包的时间戳信息进行同步就可以继续正常播放；若为非关键帧，则向前遍历或向后遍历视频流中的视频包，若遍历到包含有完整信息的关键帧的视频包时停止遍历，并从遍历后的位置继续播放。

这种解决方案不能完美解决视频进度条转为非关键帧问题，会给用户带来不良体验。这就是为什么在使用某些视频播放器的时候进度始终跳转不到想要跳转的地方的原因。

(3) 当跳转到非关键帧时，首先记录当前跳转的位置，然后在视频流中向前遍历。当遍历到一个关键帧时，将此关键帧保存，并让解码器对该帧与该帧之后的帧进行逐一解码，直至解码到刚开始记录跳转的地方。然后把视频时间戳信息与时钟进行同步并继续播放视频。

这种解决方案完美结局了非关键帧跳转的问题，用户跳转到整个视频的任意位置都可以正常加载该位置的图像并进行正常的播放。当然，这个过程在某些视频文件中关键帧较少的情况下，从关键帧计算到非关键帧的位置的时间比较长，会出现短暂的跳转后的卡顿现象，而随着处理器性能的不断提升，这种现象也逐渐被忽略。

4. 播放器设计

本文设计的视频播放器的功能共两部分：视频播放功能和视频处理功能。视频文件由于其一般占用空间较大，在处理的过程中不会将整个视频文件的数据加载至内存中，而且处理后的视频文件也不会将整个视频文件的数据放置至内存中。因此，若要对视频源文件进行处理，并尽量减少线程间的交互，需要有视频算法的插件自行解码视频文件并处理，然后将处理后的结果自行保存在硬盘中，只将保存后的文件路径作为算法处理的结果返回至主程序中。其功能结构图如图 2。

对于播放器的视频播放页面，应包含功能区、绘图区、进度控制区。功能区的作用就是向用户提供一系列的功能按钮，方便使用视频播放的功能(包括铺满窗口显示功能、原始比例显示功能、原始大小显示功能、正常播放功能、超级播放功能、按秒跳转进度功能、保存帧序列功能、按帧跳转进度功能等)。绘图区的作用就是用来显示解码后的图片，也就是播放视频。进度控制区主要包括播放暂停按钮，进度条拉动框，以及播放时间显示。

在播放器将音视频数据文件解码后，音频数据交给声卡进行播放，而图像数据则交给 Qt 的控件进行绘制显示。而在显示之前，加入了图像处理模块，在这里，用户可以选择加载自定义的图像处理算法，将图片进行实时处理，由于使用了之前提到的音视频同步机制，不会因为图像处理导致的显示速率降低而导致音视频不同步，从而视频播放模块便可以随时动态的加载图像算法并将算法处理的结果以视频播放的形式展示。

5. 解码流程设计

Ffmpeg 是一套可将音视频数据转换为流数据的开源编解码器[6]，包含了先进的解码库 libavcodec，具有音视频采集、格式转换视频截图甚至是添加水印等功能[7] [8] [9] [10]。本文使用 ffmpeg 作为解码库，

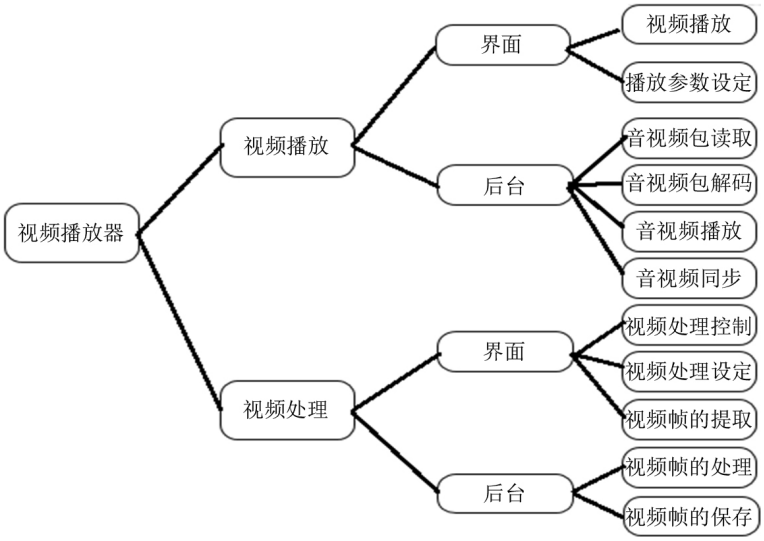

Figure 2. Functional structure
图 2. 功能结构

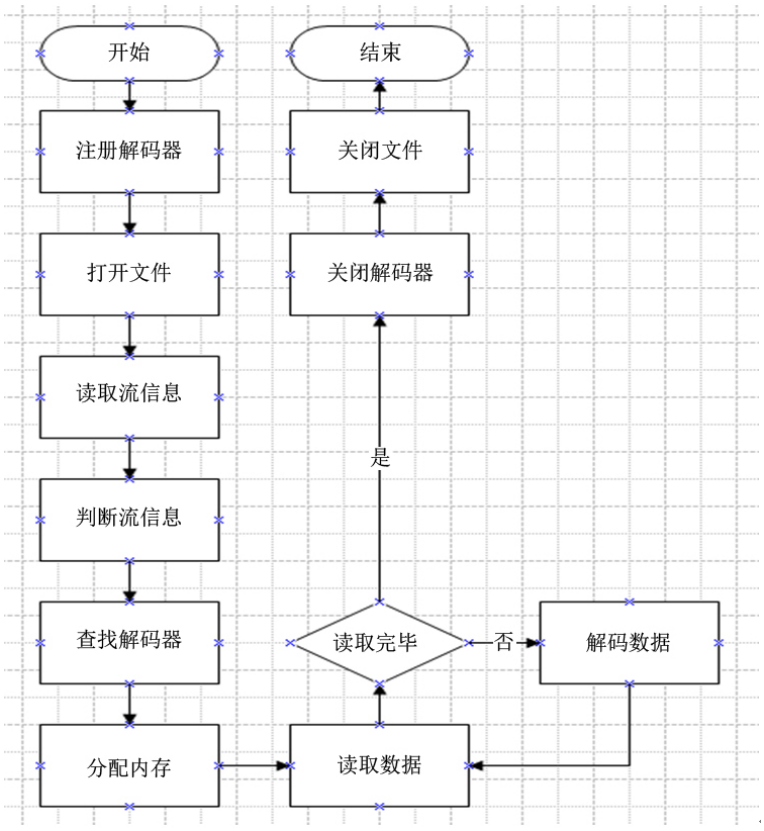

Figure 3. Decoding flow chart by ffmpeg
图 3. ffmeng 解码流程图

在 Qt 平台下完成整个解码过程[11]，从注册解码库到打开文件再到视频解码，最后再关闭文件，详细的 ffmpeg 的解码流程如下：

1.注册 ffmpeg 库中所有支持的文件格式和解码器。

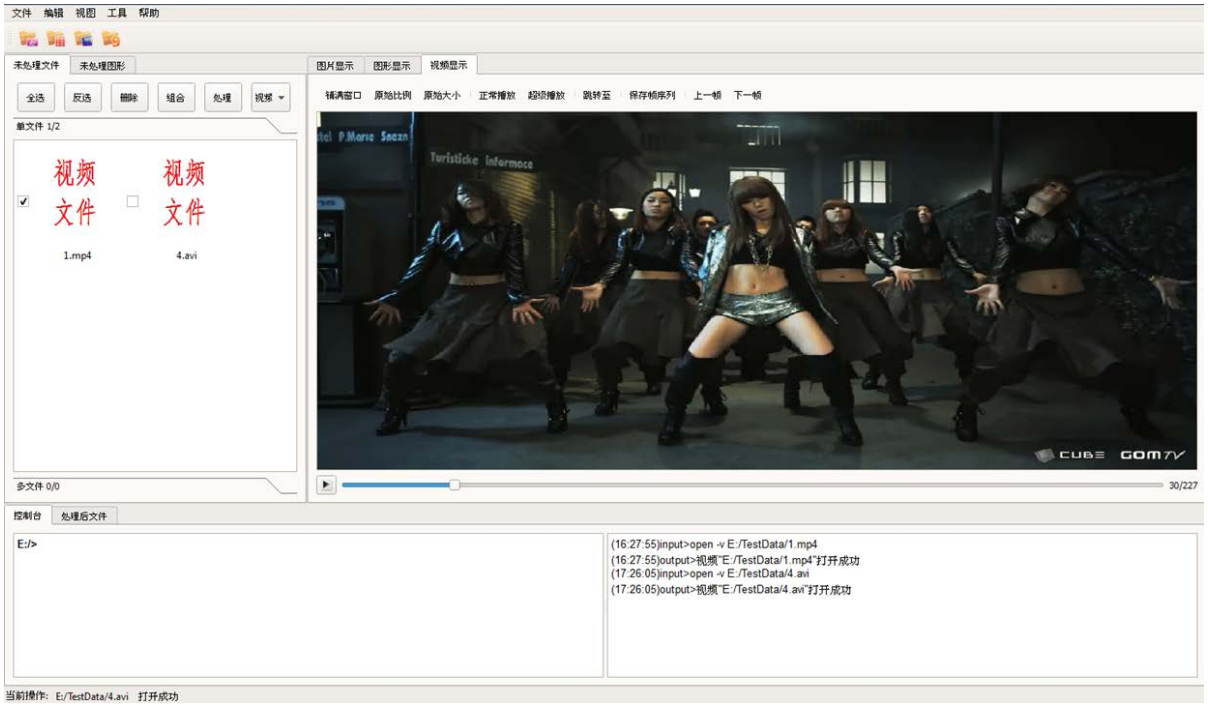

Figure 4. Player Running diagram
图 4. 播放器运行图

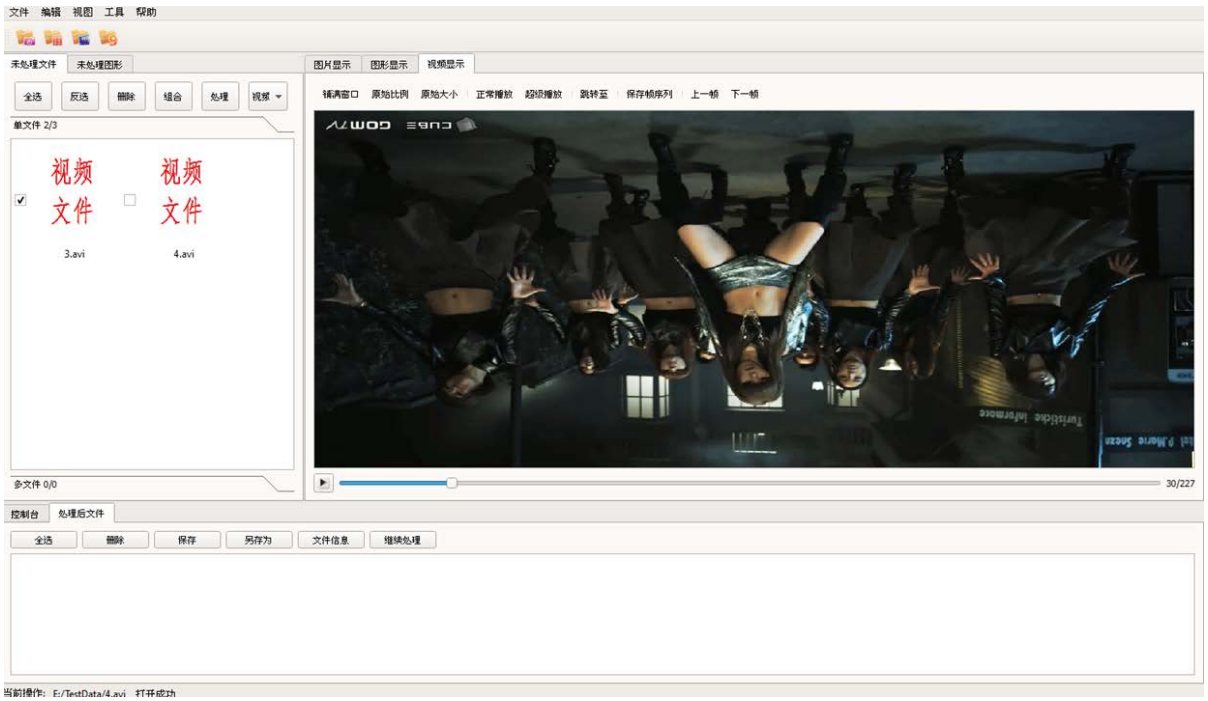

Figure 5. Player Running with loading Image Rotation algorithm
图 5. 加载图像旋转算法的播放器运行图

2.打开视频文件。

3.从视频文件中读取视频流与音频流。

4.判断音频流和视频流各自的 ID。

5.根据文件格式查找相应的解码器。

6.打开解码器。

7.为临时帧分配内存，用于保存处理中间过程。

8.循环读取视频数据中的信息。

9.对读取的数据进行解码。

10.关闭解码器。

11.关闭视频文件。

相应的流程图如图 3 所示。

6. 播放效果展示

使用本文设计的播放器控件嵌入到某平台的运行效果如图 4 所示。

除此之外，播放器还支持对图像算法的动态加载，即播放过程中同时对每一帧对象进行相应的处理，图 5 为动态加载了图像旋转算法的播放效果。

这里使用图像旋转算法仅作为测试使用，播放器使用者可以加载自定义的各类图像算法并实时进行播放。

7. 结束语

本文详细阐述了视频播放器的设计实现思路，并拓展为可动态加载图像算法的视频播放器。视频播放效果清晰流畅，视频播放功能方便于查看对比图像算法的处理效果，视频处理框架方便于算法编写人员进行图像算法的编写。具有良好的可扩展性，稳定性和可操作性。

参考文献 (References)

[1] 王建民, 张红壮. 基于 Qt 的嵌入式媒体播放器系统的设计[J]. 微计算机信息, 2008, 24(20): 64-66.

[2] Andersonyan. MPEG2-TS 音视频同步原理[J]. Chinaunix, 2012-01-20.

[3] 王少燕. 多媒体通信中的音视频同步问题研究与实现[D]: [硕士学位论文]. 西安: 西安电子科技大学, 2003.

[4] Lei, X.H., Jiang, X.H. and Wang, C.H. (2013) Design and Implementation of a Real-Time Video Stream Analysis System Based on FFMPEG. School of Information Engineering Communication University of China, Beijing. https://doi.org/10.1109/wcse.2013.38

[5] 刘马飞, 曾学文, 倪宏. Windows 平台下应用 FFMPEG 实现 H.264 视频回放[J]. 微计算机应用, 2008, 29(11): 61-65.

[6] 张达运. 基于 ffmpeg 库的嵌入式媒体播放器的开发[D]: [硕士学位论文]. 武汉: 中南民族大学, 2001.

[7] 张国庆. 基于 ffmpeg 的视频转码与保护系统的设计与实现[D]: [硕士学位论文]. 武汉:华中师范大学, 2011.

[8] Cheng, Y., Liu, Q.T., Zhao, C.L., Zhu, X.L. and Zhang, G.Q. (2012) Design and Implementation of Mediaplayer Based on FFmpeg. *Software Engineering and Knowledge Engineering*: *Theory and Practice*, **115**, 867-874. https://doi.org/10.1007/978-3-642-25349-2_114

[9] Korbel, F. (2012) FFmpeg Basics: Multimedia Handling with a Fast Audio and Video Encoder. CreateSpace Independent Publishing Platform, Charleston.

[10] Wu, Z.-S. and Zhang, X. (2006) Research and Realization of Video Coding and Storage Based on FFmpeg. *Journal of Hangzhou Dianzi University*, **3**, 391-432.

[11] 蔡志明, 卢传富, 李立夏. 精通 Qt4 编程[M]. 北京: 电子工业出版社.